Exhibit 8 to Complaint
Intellectual Ventures I LLC and Intellectual Ventures II LLC

Example American Count II Systems and Services
U.S. Patent No. 7,822,841 ("the '841 Patent")

The Accused Systems and Services include without limitation American systems and services that utilize Kubernetes; all past, current, and future systems and services that operate in the same or substantially similar manner as the specifically identified systems and services; and all past, current, and future American systems and services that have the same or substantially similar features as the specifically identified systems and services ("Example American Count II Systems and Services" or "American Systems and Services").[1]

On information and belief, the American Systems and Services use Kubernetes in public and/or private cloud(s). For example, American posts, or has posted, job opportunities that require familiarity with Kubernetes containerization concepts.

- Example of job listing for an Engineer/Sr Engineer, IT Situational Awareness at American Airlines which requires proficiency in Kubernetes. https://jobs.aa.com/job/EngineerSr-Engineer%2C-IT-Situational-Awareness/75837-en_US.
- Example of job listing for an Associate Developer, IT Applications at American Airlines which requires proficiency in Kubernetes. https://jobs.aa.com/job/Associate-Developer%2C-IT-Applications/75816-en_US.
- Example of Senior Java Full Stack Developer position at American Airlines which mentions use of Kubernetes. https://www.linkedin.com/in/rohitha-m6363/.
- Example of Sr. Kubernetes Engineer/Architect position at American Airlines which mentions heavy use of Kubernetes. https://www.linkedin.com/in/sudheer-patchari/.
- Example of Kubernetes Engineer position at American Airlines which mentions heavy use of Kubernetes. https://www.linkedin.com/in/sridhar-pulluri-199b56250/.
- Example of Sr. Cloud Infra DevSecOps Engineer/Architect position at American Airlines which mentions use of Kubernetes. https://www.linkedin.com/in/rupa-m-b90836309/.

[1] For the avoidance of doubt, Plaintiffs do not accuse the public clouds of Defendant, to the extent those services are provided by a cloud provider with a license to Plaintiffs' patents that covers Defendant's activities. Plaintiffs do not accuse the public clouds of Defendants if those services are provided by a cloud provider with a license to Plaintiffs' patents that covers Defendants' activities. Plaintiffs accuse Defendant private clouds that implement Kubernetes and non-licensed public clouds that Defendant uses to support Kubernetes for its systems and services. Plaintiffs will provide relevant license agreements for cloud providers in discovery, to the extent any such license agreements have not already been produced. To the extent any of these licenses are relevant to Defendant's activities, Plaintiffs will meet and confer with Defendant about the impact of such license(s). Once a protective order is entered into the case, Plaintiffs will provide further details.

- Example of DevOps Engineer position at American Airlines which mentions use of Kubernetes. https://www.linkedin.com/in/manidhar-a-555726169/.
- Example of Java Developer position at American Airlines which mentions use of Kubernetes.https://www.linkedin.com/in/suganya-koodalingam-8a0590102/.
- Example of Senior Java Developer position at American Airlines which mentions use of Kubernetes. https://www.linkedin.com/in/ganesh-kenda/.

As another example, American has announced cloud migration of legacy technology and efforts to modernize its mainframes and servers. Source: https://dxc.com/sg/en/insights/customer-stories/american-airlines-cloud-data-automation. American continues to use private cloud for at least certain applications. Source: https://www.techtarget.com/searchdatamanagement/feature/American-Airlines-lowers-data-management-costs-with-Intel ("American Airlines' initial target for cost optimization was Azure Data Lake, according to Vijay Premkumar, senior manager of public and *private cloud* at the airline.") (emphasis added).

On information and belief, additional information confirms American uses Kubernetes technology.

Q2: You've been leading digital transformation and modernization initiatives for a while, in very complex, demanding organizations. Looking back at your career, what are some of the challenges (organizational, business, and technical) that you encountered with modernization initiatives?

Jason: In late 2018, we started our digital transformation by creating a tenancy in Oracle Cloud Infrastructure (OCI) and iterating through multiple proofs of technology, first proving out that our on-premises applications could not only run functionally, but also run workloads as fast or even faster in OCI. Shortly after that, a core group of infrastructure team members (led by our brilliant technical delivery manager, Vijay Krishnaswamy) began learning the fundamentals of Kubernetes and related open source software. Despite their team having success migrating their applications into OCI in 2020 and then redeploying their applications into Kubernetes a year later, we faced a larger challenge with the team supporting Ventana, the core application that supports the AAdvantage program. The stakes were huge: Ventana is one of American Airlines' most important applications. Additionally, the team supporting it had not been directly involved in the journey forged by the other group.

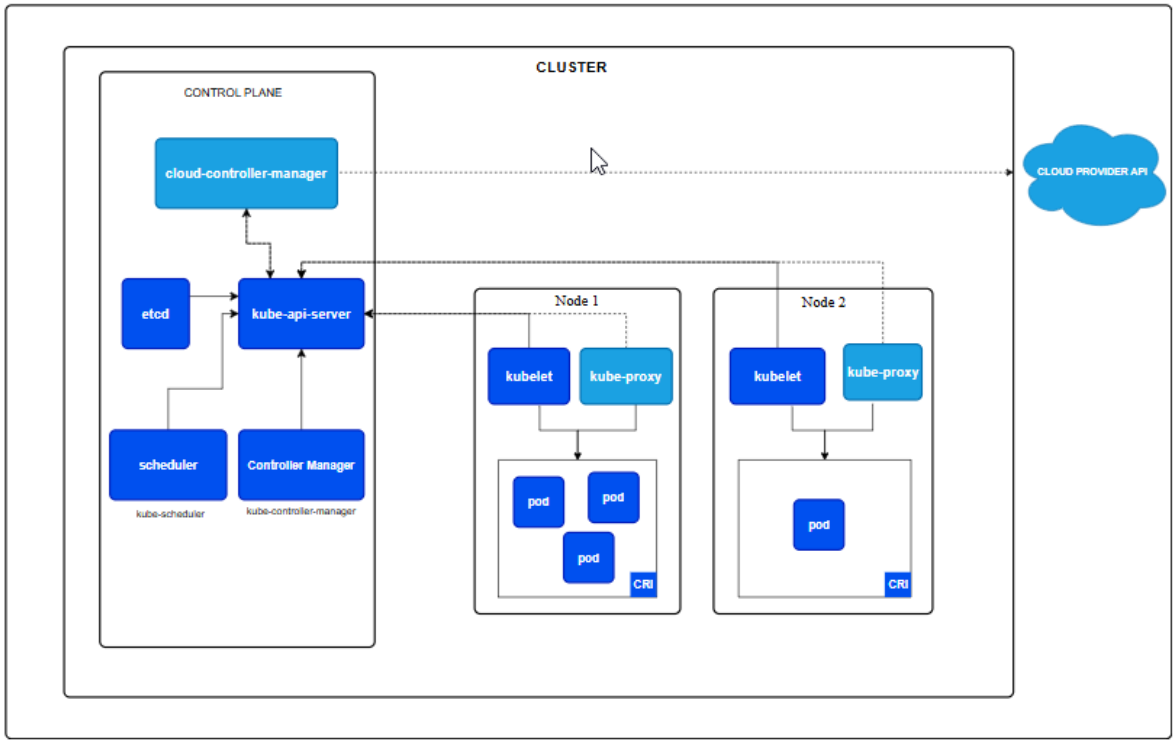
When we merged both teams into one organization, encouraged team members to learn from and teach each other, and demonstrated sustained leadership support, we emerged with a crossfunctional team who achieved multiple certifications in both OCI and Kubernetes. They were then equipped with the skills, confidence, and support to move Ventana into not just OCI, but also Oracle Container Engine for Kubernetes (OKE) in one go.

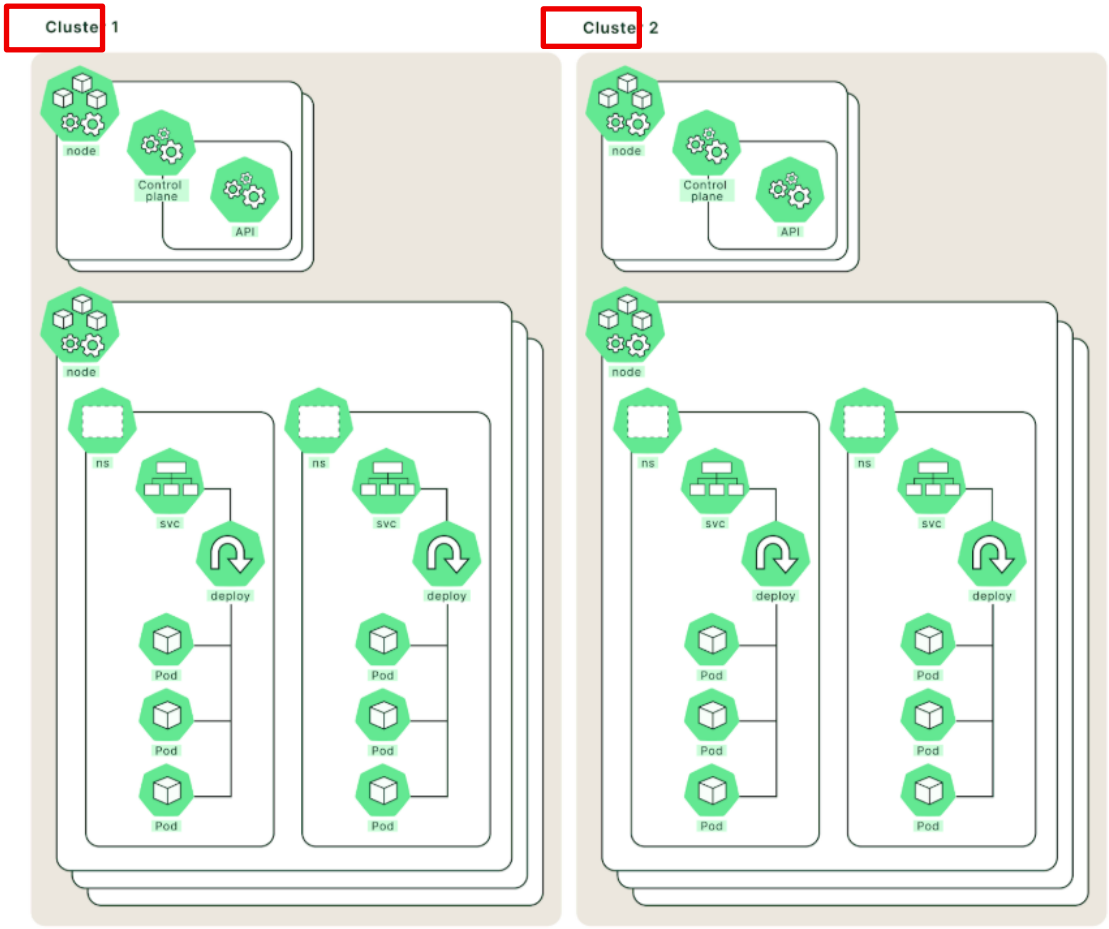
Source: https://blogs.oracle.com/cloud-infrastructure/post/five-questions-jason-maczura-american-airlines.[2]

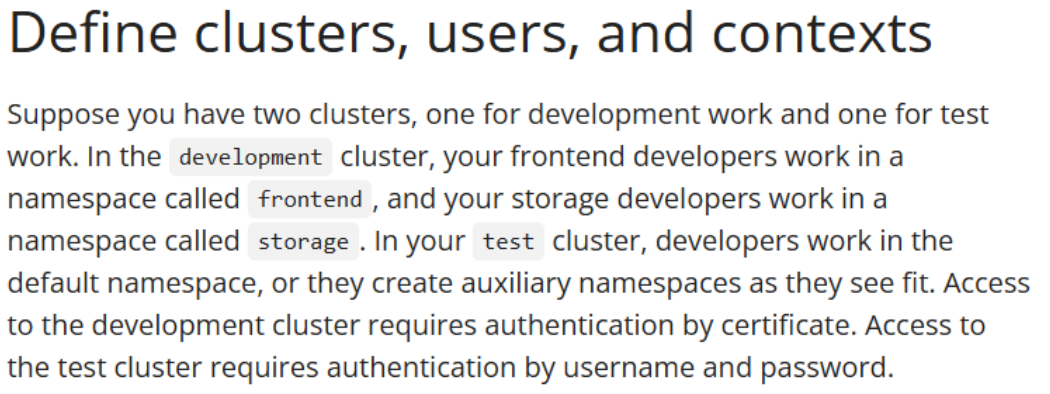
[2] Unless otherwise noted, all sources cited in this document were publicly accessible as of the date of the Complaint.

U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
1[Pre] A computer system for hosting computing clusters for clients, comprising:	To the extent this preamble is limiting, on information and belief, the American Count II Systems and Services utilizes a computer system capable of hosting computing clusters for clients. Kubernetes deployed on a server system is a portable, extensible platform for managing containerized workloads and services. **Overview** Kubernetes is a portable, extensible, open source platform for managing containerized workloads and services, that facilitates both declarative configuration and automation. It has a large, rapidly growing ecosystem. Kubernetes services, support, and tools are widely available. Source: https://kubernetes.io/docs/concepts/overview/. Further, Kubernetes uses clusters, which are groups of nodes (for example, node 1 and node 2) that host and run containerized applications as per their defined deployments and services.

U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	Cluster Architecture The architectural concepts behind Kubernetes. A Kubernetes cluster consists of a control plane plus a set of worker machines, called nodes, that run containerized applications. Every cluster needs at least one worker node in order to run Pods. The worker node(s) host the Pods that are the components of the application workload. The control plane manages the worker nodes and the Pods in the cluster. In production environments, the control plane usually runs across multiple computers and a cluster usually runs multiple nodes, providing fault-tolerance and high availability. Source: https://kubernetes.io/docs/concepts/architecture/.

U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	 Figure 1. Kubernetes cluster components. Source: https://kubernetes.io/docs/concepts/architecture/. As shown below, Kubernetes supports multiple clusters, which can be hosted on a computer system. Kubernetes is designed to scale quickly, massively, and reliably. However, scaling in place — such as adding more nodes to a cluster — eventually reaches a point of diminishing returns. Kubernetes multi-cluster, which is multiple Kubernetes clusters operating as a single logical platform, helps solve this problem and enhances Kubernetes capabilities in many critical areas. Source: https://www.kubecost.com/kubernetes-multi-cloud/kubernetes-multi-cluster/.

U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	 Source: https://www.kubecost.com/kubernetes-multi-cloud/kubernetes-multi-cluster/.

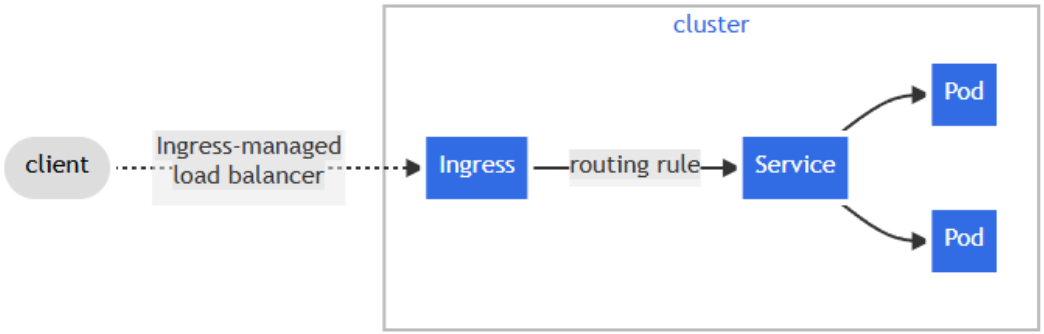
U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	 Source: https://kubernetes.io/docs/tasks/access-application-cluster/configure-access-multiple-clusters/.
1[a] a private communications network linked to a public communications network;	Based on information and belief, the American Count II Systems and Services includes a private communications network linked to a public communications network. For example, the Kubernetes system within the American accused network is configured to be linked through, for example, an Ingress, to an external network such as the Internet.

U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	# Terminology For clarity, this guide defines the following terms: • Node: A worker machine in Kubernetes, part of a cluster. • Cluster: A set of Nodes that run containerized applications managed by Kubernetes. For this example, and in most common Kubernetes deployments, nodes in the cluster are not part of the public internet. • Edge router: A router that enforces the firewall policy for your cluster. This could be a gateway managed by a cloud provider or a physical piece of hardware. • Cluster network: A set of links, logical or physical, that facilitate communication within a cluster according to the Kubernetes networking model. • Service: A Kubernetes Service that identifies a set of Pods using label selectors. Unless mentioned otherwise, Services are assumed to have virtual IPs only routable within the cluster network. Source: https://kubernetes.io/docs/concepts/services-networking/ingress/. ## The Kubernetes network model …

U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	• The *pod network* (also called a cluster network) handles communication between pods. It ensures that (barring intentional network segmentation): ◦ All pods can communicate with all other pods, whether they are on the same node or on different nodes. Pods can communicate with each other directly, without the use of proxies or address translation (NAT). On Windows, this rule does not apply to host-network pods. ◦ Agents on a node (such as system daemons, or kubelet) can communicate with all pods on that node. Source: https://kubernetes.io/docs/concepts/services-networking/. Further, in Kubernetes, Services provide a way for communication within the American accused network to an external network, such as the Internet. For example, the Ingress is configured to expose HTTP/HTTPS routes from outside the cluster. In Kubernetes, you generally need to use a Service to expose an application in your cluster to the internet. A service groups together pods performing the same function (e.g. running the same application) and defines how to access them. The most basic type of service is **clusterIP**, but this only provides internal access, from within the cluster, to the defined pods. The **NodePort** and **LoadBalancer** services both provide external access. **Ingress** (which is not a service but an API object inside a cluster) combined with an explicitly-created **ingress controller** is another way to expose the cluster. Source: https://www.scaleway.com/en/docs/kubernetes/reference-content/exposing-services/.

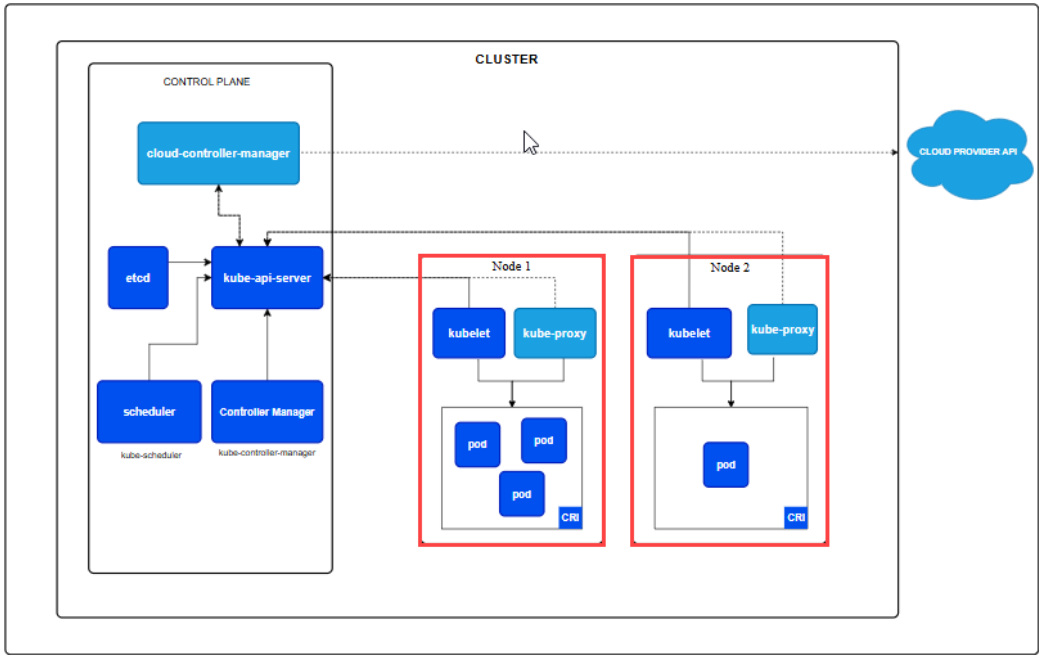
U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	Service Expose an application running in your cluster behind a single outward-facing endpoint, even when the workload is split across multiple backends. In Kubernetes, a Service is a method for exposing a network application that is running as one or more Pods in your cluster. Source: https://kubernetes.io/docs/concepts/services-networking/service/. A key aim of Services in Kubernetes is that you don't need to modify your existing application to use an unfamiliar service discovery mechanism. You can run code in Pods, whether this is a code designed for a cloud-native world, or an older app you've containerized. You use a Service to make that set of Pods available on the network so that clients can interact with it. Source: https://kubernetes.io/docs/concepts/services-networking/service/.

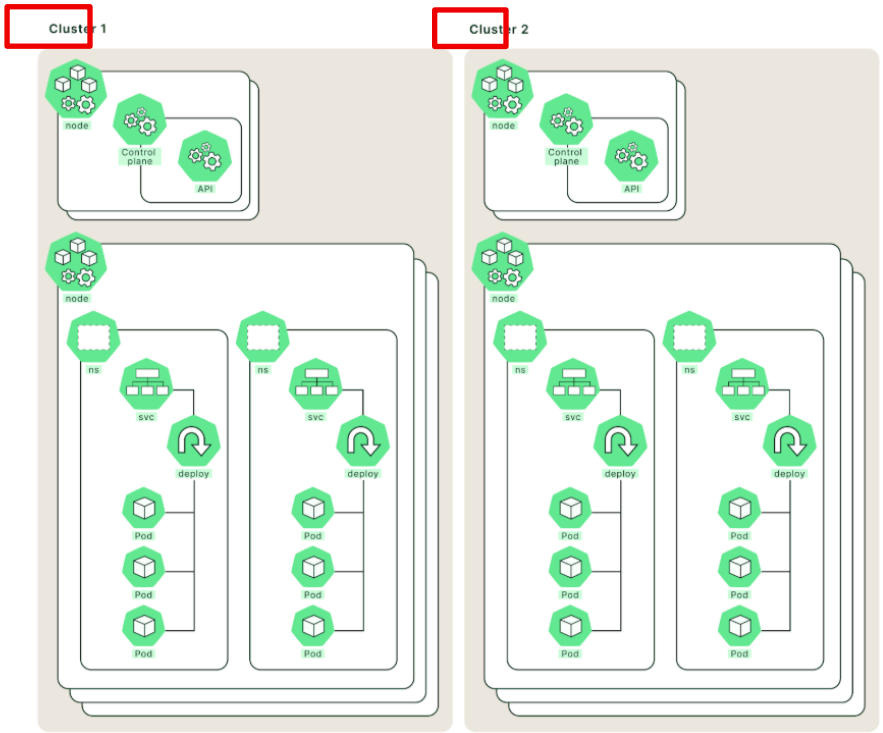
U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	# Services in Kubernetes The Service API, part of Kubernetes, is an abstraction to help you expose groups of Pods over a network. Each Service object defines a logical set of endpoints (usually these endpoints are Pods) along with a policy about how to make those pods accessible. Source: https://kubernetes.io/docs/concepts/services-networking/service/. If your workload speaks HTTP, you might choose to use an Ingress to control how web traffic reaches that workload. Ingress is not a Service type, but it acts as the entry point for your cluster. An Ingress lets you consolidate your routing rules into a single resource, so that you can expose multiple components of your workload, running separately in your cluster, behind a single listener. Source: https://kubernetes.io/docs/concepts/services-networking/service/. Ingress exposes HTTP and HTTPS routes from outside the cluster to services within the cluster. Traffic routing is controlled by rules defined on the Ingress resource. Source: https://kubernetes.io/docs/concepts/services-networking/ingress/.

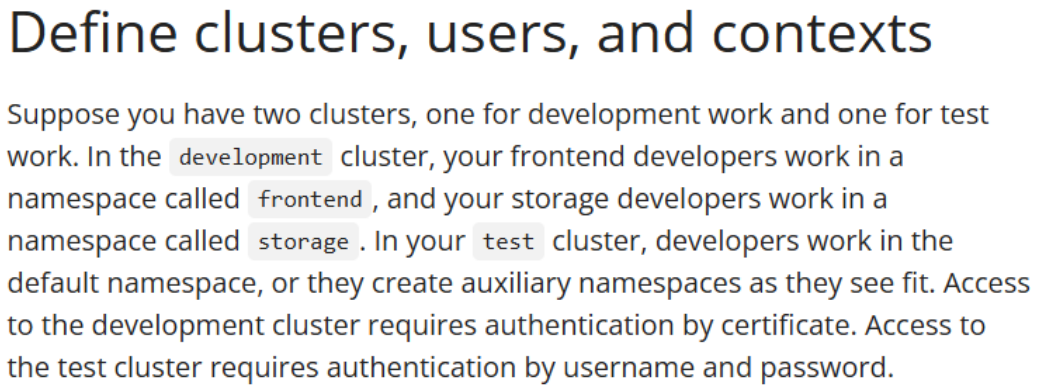
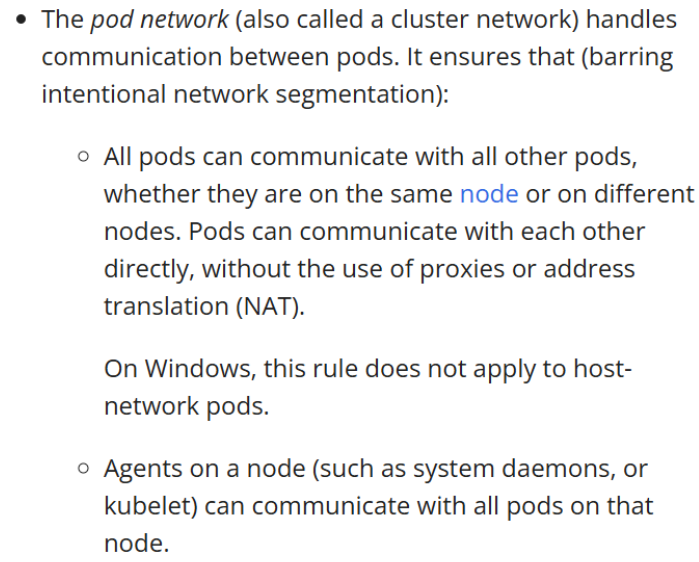
U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	 Figure. Ingress Source: https://kubernetes.io/docs/concepts/services-networking/ingress/. A Kubernetes system from within the American accused network includes the Gateway API, which provides Gateway resources. These resources can be defined and specified so that external traffic from the Internet can be routed to various Services. • The Gateway API (or its predecessor, Ingress) allows you to make Services accessible to clients that are outside the cluster. Source: https://kubernetes.io/docs/concepts/services-networking/. The Gateway API for Kubernetes provides extra capabilities beyond Ingress and Service. You can add Gateway to your cluster - it is a family of extension APIs, implemented using CustomResourceDefinitions - and then use these to configure access to network services that are running in your cluster. Source: https://kubernetes.io/docs/concepts/services-networking/service/.

U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	Resource model Gateway API has three stable API kinds: • **GatewayClass:** Defines a set of gateways with common configuration and managed by a controller that implements the class. • **Gateway:** Defines an instance of traffic handling infrastructure, such as cloud load balancer. • **HTTPRoute:** Defines HTTP-specific rules for mapping traffic from a Gateway listener to a representation of backend network endpoints. These endpoints are often represented as a Service. Source: https://kubernetes.io/docs/concepts/services-networking/gateway/.  In this example, the request flow for a Gateway implemented as a reverse proxy is: 1. The client starts to prepare an HTTP request for the URL `http://www.example.com` Source: https://kubernetes.io/docs/concepts/services-networking/gateway/.

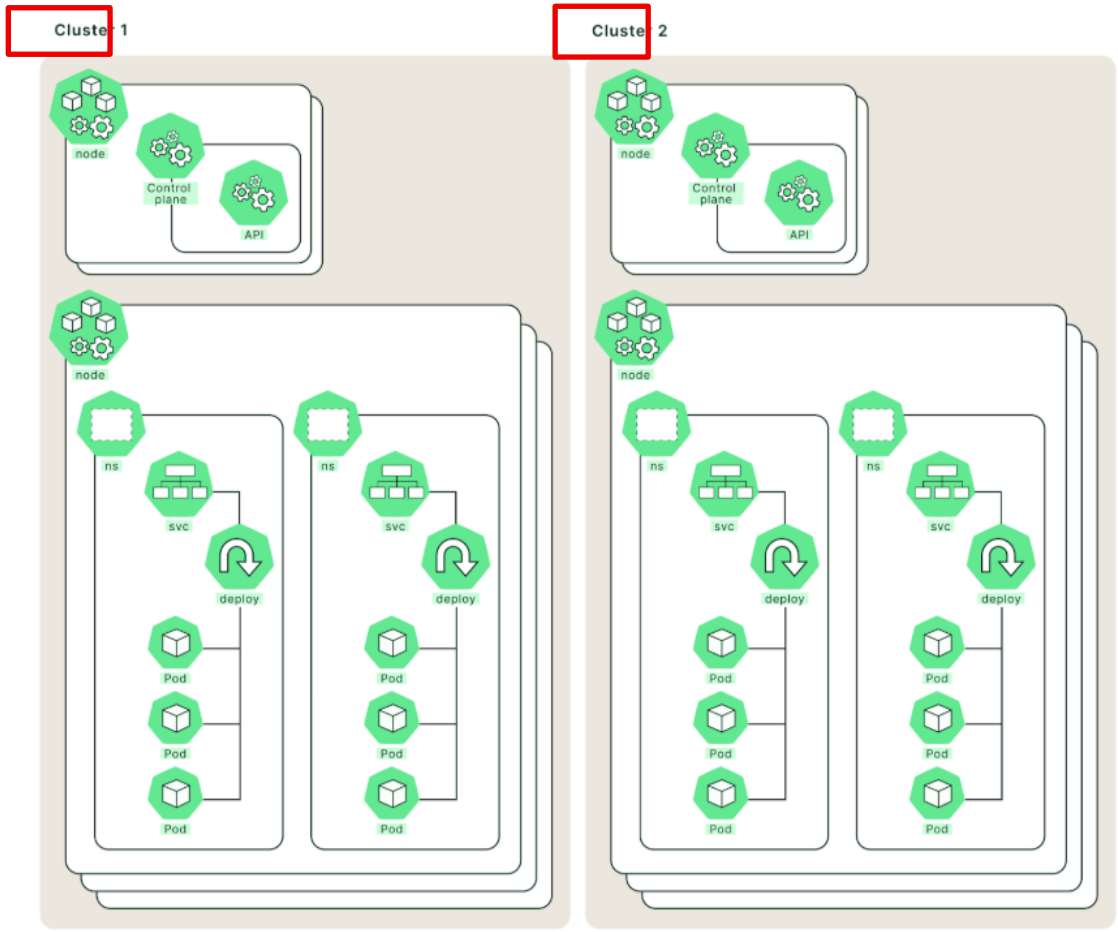
U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
1[b] a first cluster comprising a set of computing resources, including at least one hardware processor, in a first configuration, wherein the first cluster is communicatively linked to the private communications network;	Based on information and belief, the American Count II Systems and Services include a first cluster comprising a set of computing resources, including at least one hardware processor, in a first configuration, wherein the first cluster is communicatively linked to the private communications network. For example, Kubernetes supports a Kubernetes cluster network to facilitate communication amongst nodes and/or Pods within the American accused network. For example, Kubernetes clusters include at least one node, where a node is either a physical or virtual machine comprising a CPU or portion of CPU resources and memory, to run workloads. These clusters are connected to a private network, for example, the American accused network. On information and belief, the private network, for example, the American accused network, facilitates container-to-container, Pod-to-Pod, and/or Pod-to-Services communications across multiple clusters for sharing data and handling tasks. **Cluster Architecture** The architectural concepts behind Kubernetes. A Kubernetes cluster consists of a control plane plus a set of worker machines, called nodes, that run containerized applications. Every cluster needs at least one worker node in order to run Pods. The worker node(s) host the Pods that are the components of the application workload. The control plane manages the worker nodes and the Pods in the cluster. In production environments, the control plane usually runs across multiple computers and a cluster usually runs multiple nodes, providing fault-tolerance and high availability. Source: https://kubernetes.io/docs/concepts/architecture/.

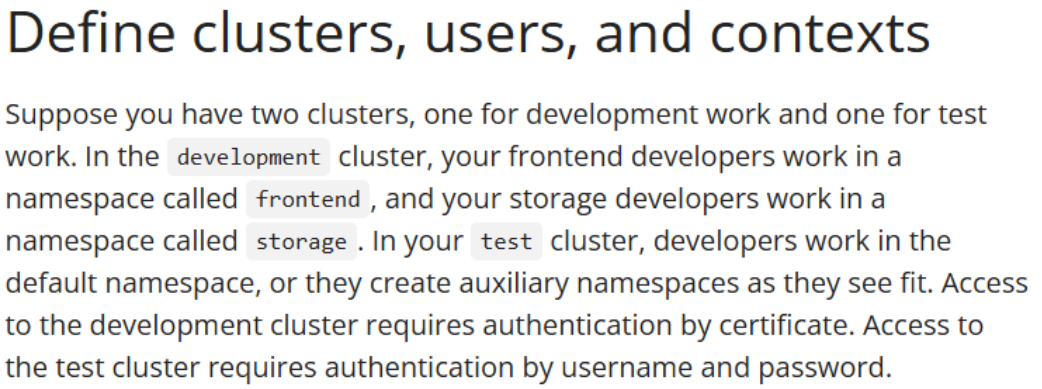
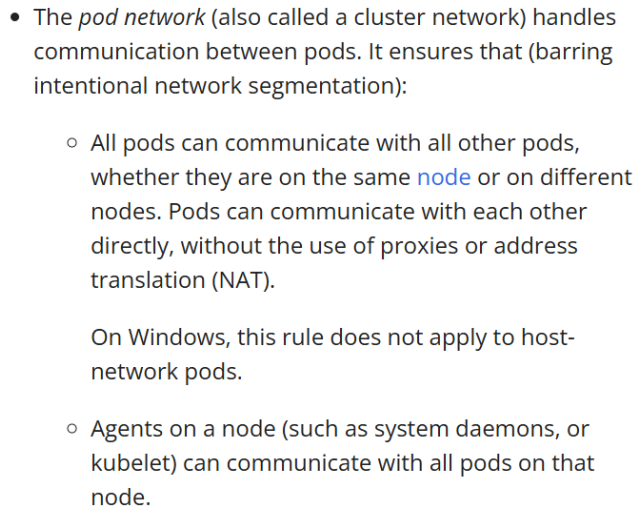
U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	 Figure 1. Kubernetes cluster components. Source: https://kubernetes.io/docs/concepts/architecture/. **What Is A Kubernetes Node?** A Kubernetes node is either a virtual or physical machine that one or more Kubernetes pods run on. It is a worker machine that contains the necessary services to run pods, including the CPU and memory resources they need to run. Source: https://www.cloudzero.com/blog/kubernetes-node-vs-pod/.

U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	Additionally, Kubernetes supports multiple clusters, which can be hosted on a computer system. Kubernetes is designed to scale quickly, massively, and reliably. However, scaling in place — such as adding more nodes to a cluster — eventually reaches a point of diminishing returns. Kubernetes multi-cluster, which is multiple Kubernetes clusters operating as a single logical platform, helps solve this problem and enhances Kubernetes capabilities in many critical areas. Source: https://www.kubecost.com/kubernetes-multi-cloud/kubernetes-multi-cluster/.  Source: https://www.kubecost.com/kubernetes-multi-cloud/kubernetes-multi-cluster/.

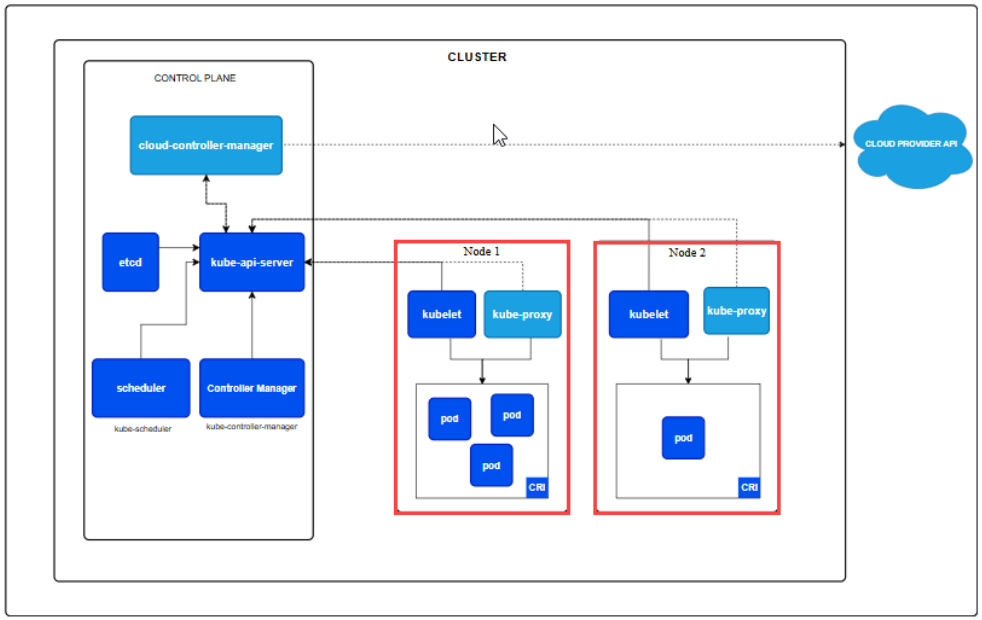
U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	 Source: https://kubernetes.io/docs/tasks/access-application-cluster/configure-access-multiple-clusters/.  Source: https://kubernetes.io/docs/concepts/services-networking/.

U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
1[c] a second cluster comprising a set of computing resources, including at least one hardware processor, in a second configuration, wherein the second cluster is communicatively linked to the private communications network; and	Based on information and belief, the American Count II Systems and Services includes a second cluster comprising a set of computing resources, including at least one hardware processor, in a second configuration, wherein the second cluster is communicatively linked to the private communications network. Kubernetes supports multiple clusters, which can be hosted on a computer system. > Kubernetes is designed to scale quickly, massively, and reliably. However, scaling in place — such as adding more nodes to a cluster — eventually reaches a point of diminishing returns. Kubernetes multi-cluster, which is multiple Kubernetes clusters operating as a single logical platform, helps solve this problem and enhances Kubernetes capabilities in many critical areas. Source: https://www.kubecost.com/kubernetes-multi-cloud/kubernetes-multi-cluster/.

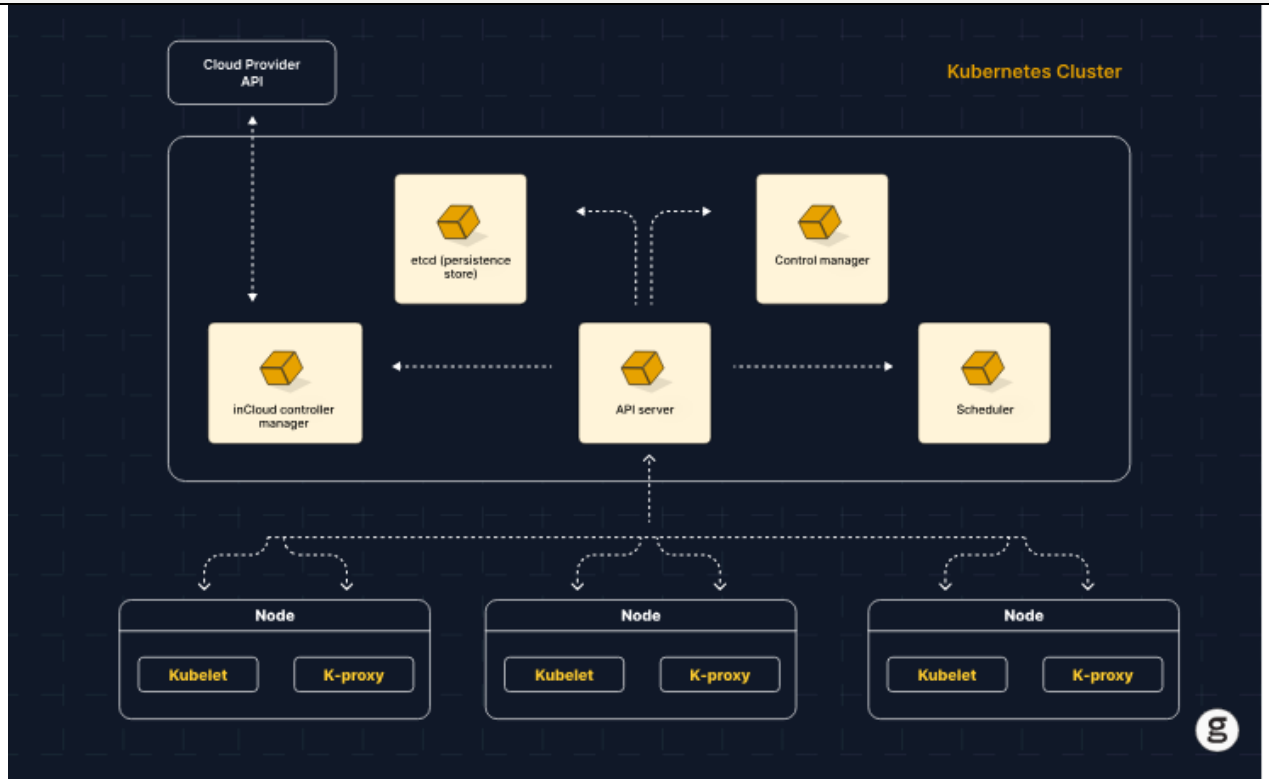
U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	 Source: https://www.kubecost.com/kubernetes-multi-cloud/kubernetes-multi-cluster/.

U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	 Source: https://kubernetes.io/docs/tasks/access-application-cluster/configure-access-multiple-clusters/.  Source: https://kubernetes.io/docs/concepts/services-networking/.

U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	For example, Kubernetes clusters include at least one node, where a node is either a physical or virtual machine comprising a CPU or portion of CPU resources and memory to run workloads. These clusters are connected to a private network, for example, the American accused network. On information and belief, the private network, for example, the American accused network, facilitates container-to-container, Pod-to-Pod, and/or Pod-to-Services communications across multiple clusters for sharing data and handling tasks. The cluster (for example second cluster) includes various nodes, wherein each node consists of various pods, and each pod further includes containers. Kubernetes supports communication amongst pods, whether they are on the same node or on different nodes. Based on information and belief, the cluster is communicatively linked to the private communication network. **Cluster Architecture** The architectural concepts behind Kubernetes. A Kubernetes cluster consists of a control plane plus a set of worker machines, called nodes, that run containerized applications. Every cluster needs at least one worker node in order to run Pods. The worker node(s) host the Pods that are the components of the application workload. The control plane manages the worker nodes and the Pods in the cluster. In production environments, the control plane usually runs across multiple computers and a cluster usually runs multiple nodes, providing fault-tolerance and high availability. Source: https://kubernetes.io/docs/concepts/architecture/.

U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	 Figure 1. Kubernetes cluster components. Source: https://kubernetes.io/docs/concepts/architecture/. What Is A Kubernetes Node? A Kubernetes node is either a virtual or physical machine that one or more Kubernetes pods run on. It is a worker machine that contains the necessary services to run pods, including the CPU and memory resources they need to run. Source: https://www.cloudzero.com/blog/kubernetes-node-vs-pod/.

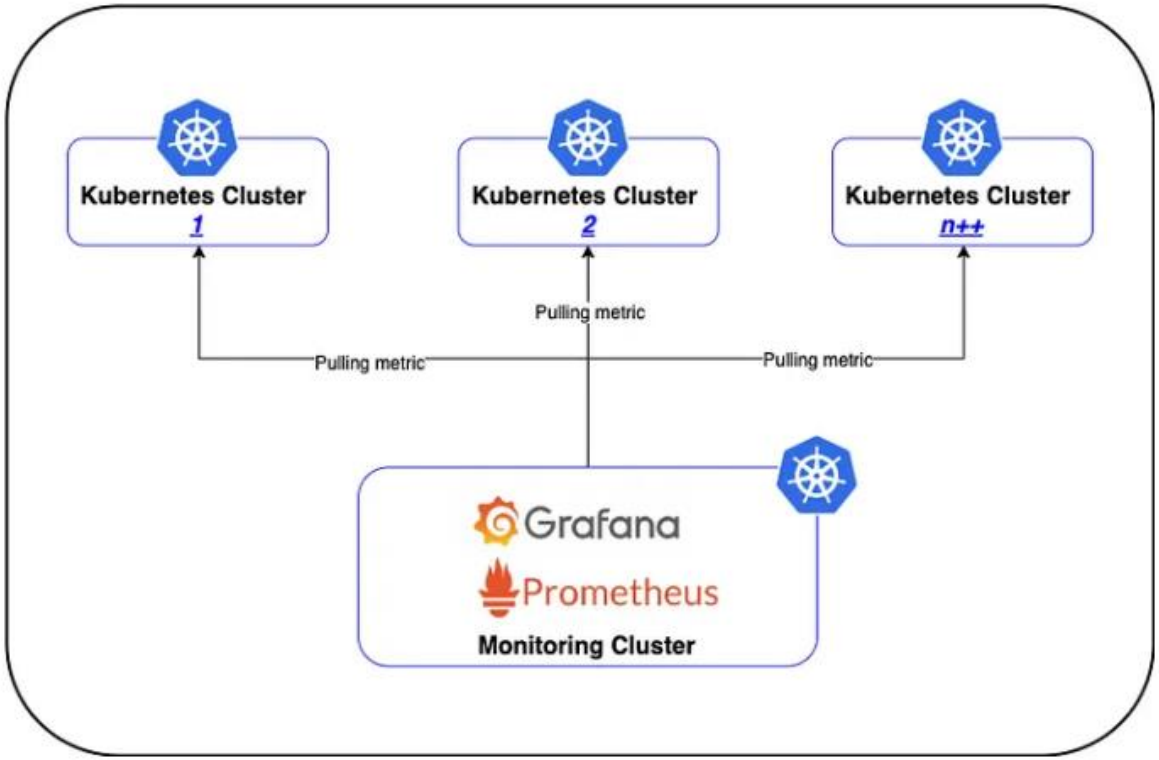
U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
1[d] a monitoring system monitoring operations of the first and second clusters, identifying operational and connectivity problems, and issuing an alert in response to the identified problems indicating a corresponding one of the first and second clusters associated with the identified problems;	Based on information and belief, the American Count II Systems and Services include a monitoring system, monitoring operations of the first and second clusters, identifying operational and connectivity problems, and issuing an alert in response to the identified problems indicating a corresponding one of the first and second clusters associated with the identified problems. Kubernetes supports a Monitoring system (via Kubernetes API) with the functionality to monitor and manage Kubernetes clusters, such as cluster's health, performance and identifying associated problems. **What is Kubernetes monitoring?** Simply put, Kubernetes monitoring is the practice of tracking the status of all components of a Kubernetes environment. Because there are many pieces inside Kubernetes, Kubernetes monitoring actually entails monitoring many distinct things, such as:

U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	 Source: https://www.groundcover.com/kubernetes-monitoring. Also, Kubernetes provides a full metrics pipeline and monitoring pipeline feature that provides monitoring of operations within Kubernetes clusters.

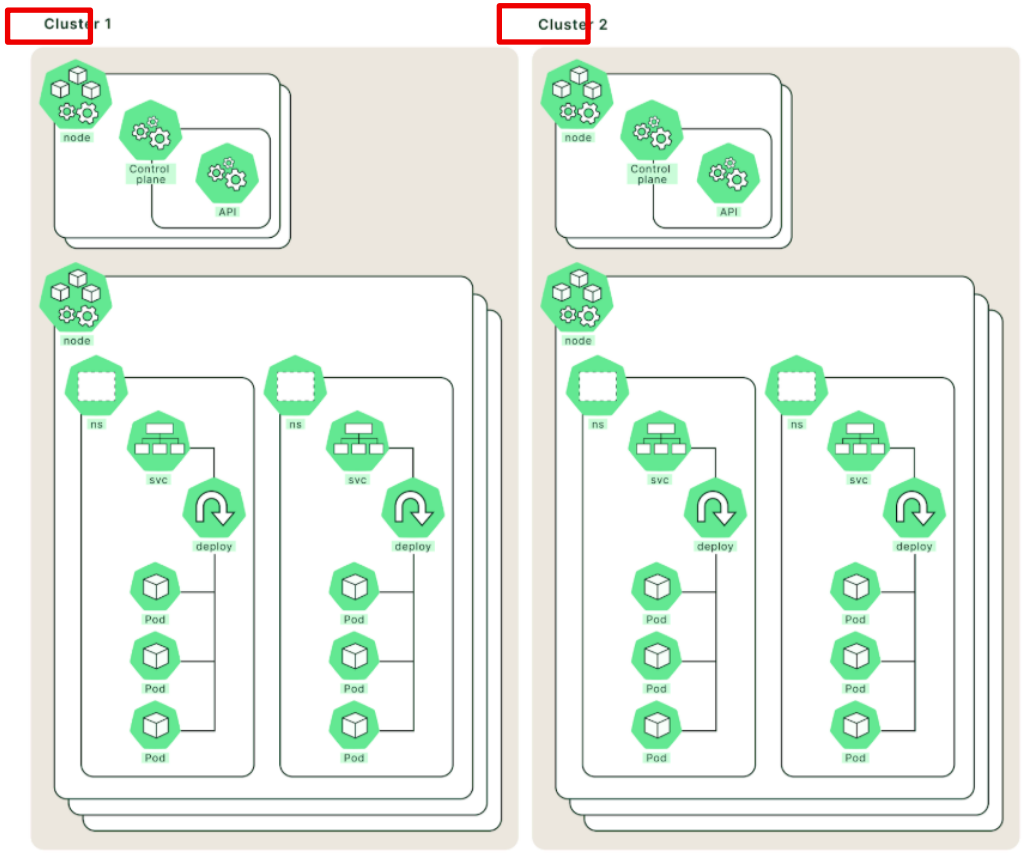
U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	A full metrics pipeline gives you access to richer metrics. Kubernetes can respond to these metrics by automatically scaling or adapting the cluster based on its current state, using mechanisms such as the Horizontal Pod Autoscaler. The monitoring pipeline fetches metrics from the kubelet and then exposes them to Kubernetes via an adapter by implementing either the `custom.metrics.k8s.io` or `external.metrics.k8s.io` API. Kubernetes is designed to work with OpenMetrics, which is one of the CNCF Observability and Analysis - Monitoring Projects, built upon and carefully extending Prometheus exposition format in almost 100% backwards-compatible ways. Source: https://kubernetes.io/docs/tasks/debug/debug-cluster/resource-usage-monitoring/. **What is Kubernetes monitoring?** Simply put, Kubernetes monitoring is the practice of tracking the status of all components of a Kubernetes environment. Because there are many pieces inside Kubernetes, Kubernetes monitoring actually entails monitoring many distinct things, such as: • The kube-system workloads. • Cluster information using the Kubernetes API. • Applications interactions with Kubernetes by monitoring apps bottom-up. Source: https://www.groundcover.com/kubernetes-monitoring.

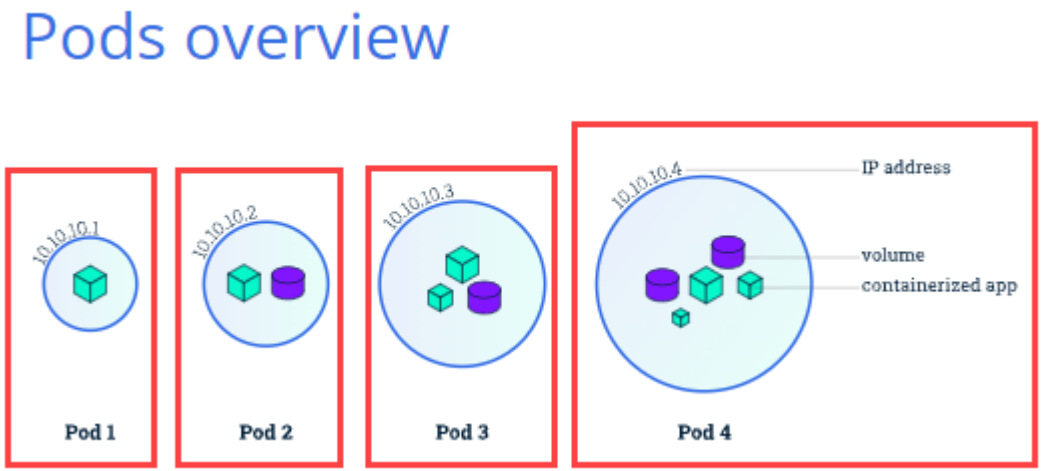
U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	Further, Kubernetes monitoring includes providing alerts to problems occurring within a cluster that enables troubleshooting and addressing other potential issues. By collecting Kubernetes data, you'll get viable information regarding your Kubernetes cluster health, that can help you perform Kubernetes troubleshooting and manage issues like unexpected container termination. You can also leverage the data for proactive decisions such as adjusting rate limits. Source: https://www.groundcover.com/kubernetes-monitoring. Kubernetes alerting is the practice of generating notifications for events or trends in Kubernetes that require admins' attention. Examples of such events and trends include: • A node that has failed. • A Pod that is stuck in the pending state. • A container or Pod that is consuming a high level of resources relative to normal consumption trends. • High latency rates for communication between Kubernetes cluster components (such as between kubelet and control plane nodes). Source: https://www.groundcover.com/kubernetes-monitoring/kubernetes-alerting. Further, Kubernetes using API for cluster monitoring and kubelets for monitoring at node level, Kubernetes also supports additional third-party open source monitoring tools such as, Grafana, Prometheus, and/or FluentD.

U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	Setting up monitoring in Kubernetes Monitoring is a crucial aspect of effectively managing Kubernetes clusters. By setting up comprehensive monitoring, you can gain insights into your cluster's health, performance, and resource utilization, enabling you to address issues and optimize your infrastructure proactively. Several monitoring solutions are available for Kubernetes, including open-source tools like Prometheus, Grafana, and Alertmanager and commercial options like Datadog and New Relic. Choose a solution that aligns with your requirements and budget. Setting up Prometheus and Grafana for monitoring your Kubernetes cluster is a common choice and can provide a robust monitoring solution. The following are detailed steps for setting up Prometheus, Grafana, and FluentD, three common monitoring components, assuming that a Kubernetes cluster is already up and running: ▪ **Prometheus:** A monitoring tool built for aggregating metrics in containers that can be used for setting up alerts. ▪ **Grafana:** A monitoring tool that can be used to visualize Prometheus data. ▪ **FluentD:** A tool for aggregating a system's logs into one source. Source: https://www.apptio.com/topics/kubernetes/monitoring/.

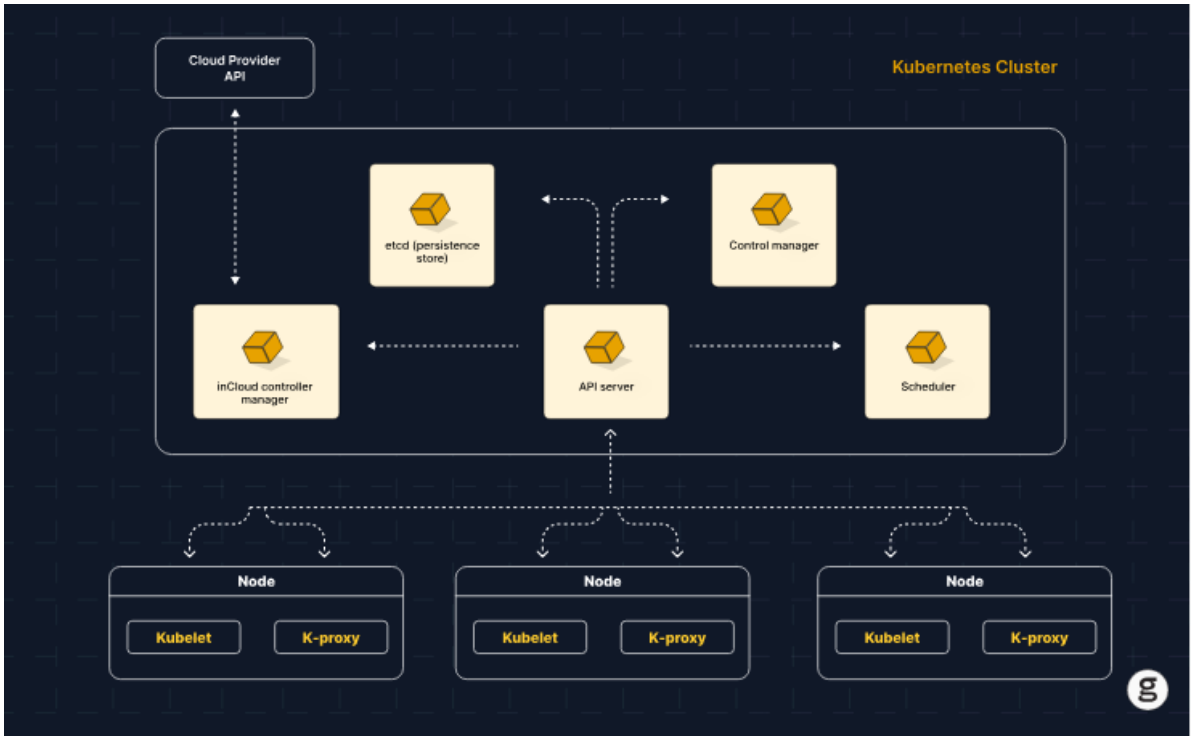
U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	 Kubernetes Cluster Monitoring Source: https://towardsdev.com/how-to-monitor-kubernetes-cluster-using-prometheus-grafana-f220b0284e65.

U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
[1.e] wherein the first configuration differs from the second configuration and wherein the first configuration provides a first computing environment for performing a first client task and the second configuration provides a second computing environment for performing a second client task;	Based on information and belief, the American Count II Systems and Services include a computer system where the first configuration differs from the second configuration and where the first configuration provides a first computing environment for performing a first client task and the second configuration provides a second computing environment for performing a second client task. Kubernetes comprises pods, which are distributed across different nodes in different clusters. These pods can be configured to perform different tasks. Each node contains services necessary to run a pod, and each pod runs its own instance of a given application container or set of application containers. Based on information and belief, the resulting configurations can be different at each node and cluster.

U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	 Source: https://www.kubecost.com/kubernetes-multi-cloud/kubernetes-multi-cluster/.

U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	Pods *Pods* are the smallest deployable units of computing that you can create and manage in Kubernetes. A *Pod* (as in a pod of whales or pea pod) is a group of one or more containers, with shared storage and network resources, and a specification for how to run the containers. A Pod's contents are always co-located and co-scheduled, and run in a shared context. A Pod models an application-specific "logical host": it contains one or more application containers which are relatively tightly coupled. In non-cloud contexts, applications executed on the same physical or virtual machine are analogous to cloud applications executed on the same logical host. Source: https://kubernetes.io/docs/concepts/workloads/pods/.  Source: https://kubernetes.io/docs/tutorials/kubernetes-basics/explore/explore-intro/.

U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	Workload resources for managing pods Usually you don't need to create Pods directly, even singleton Pods. Instead, create them using workload resources such as Deployment or Job. If your Pods need to track state, consider the StatefulSet resource. Each Pod is meant to run a single instance of a given application. If you want to scale your application horizontally (to provide more overall resources by running more instances), you should use multiple Pods, one for each instance. In Kubernetes, this is typically referred to as *replication*. Replicated Pods are usually created and managed as a group by a workload resource and its controller. Source: https://kubernetes.io/docs/concepts/workloads/pods/.
[1.f] wherein the monitoring system comprises a main monitor that operates to monitor the first and second clusters to identify the operation and connectivity problems and further comprises monitors for each node of the first and second clusters operating to check for hardware and software problems within a particular node and to report the hardware and software problems to the main monitor.	Based on information and belief, the American Count II Systems and Services include a computer system where the monitoring system comprises a main monitor that operates to monitor the first and second clusters to identify the operation and connectivity problems and further comprises monitors for each node of the first and second clusters operating to check for hardware and software problems within a particular node and to report the hardware and software problems to the main monitor. Kubernetes supports a Monitoring system (via Kubernetes API) with the functionality to monitor and manage Kubernetes clusters such as cluster's health, performance and identifying associated problems.

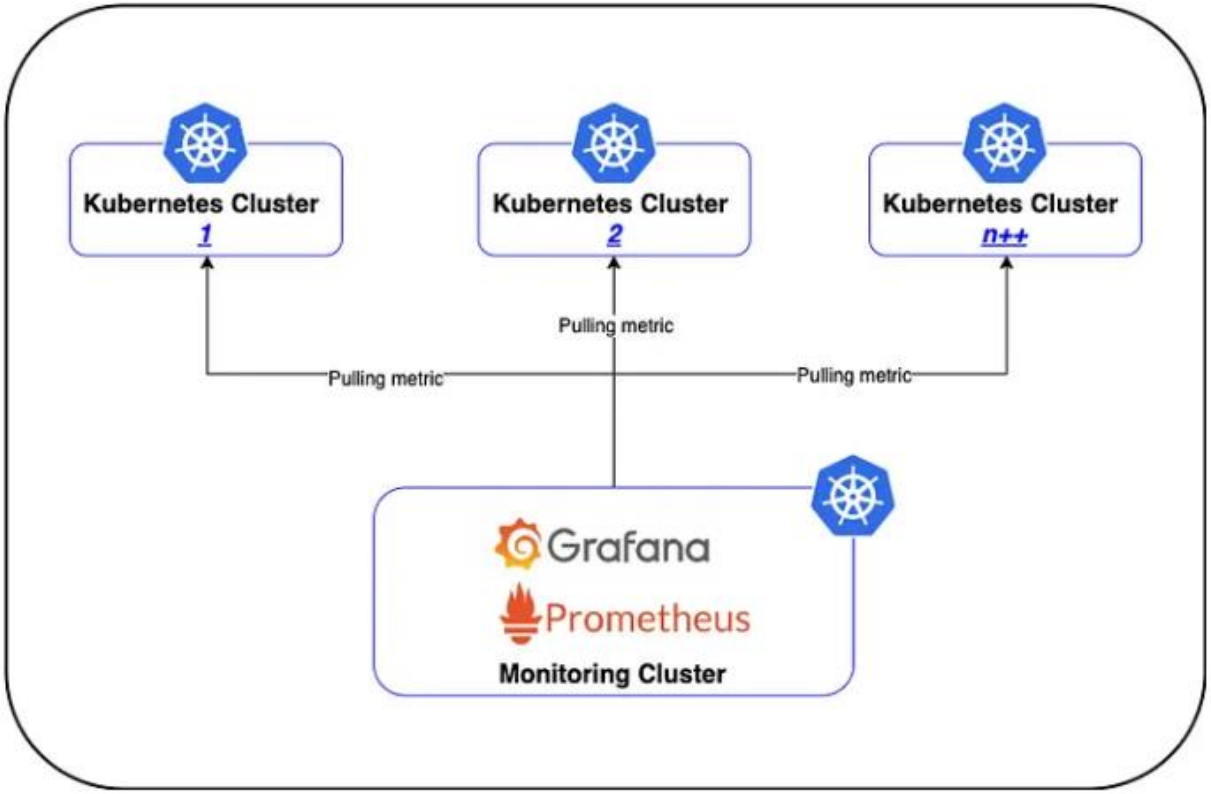
U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	What is Kubernetes monitoring? Simply put, Kubernetes monitoring is the practice of tracking the status of all components of a Kubernetes environment. Because there are many pieces inside Kubernetes, Kubernetes monitoring actually entails monitoring many distinct things, such as:  Source: Kubernetes Monitoring 101: Challenges & Best Practices.

U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	Kubernetes works with metrics pipeline/monitoring pipeline features, that provide monitoring operations within Kubernetes clusters including monitoring software and hardware problems. Kubernetes supports monitoring functionalities for containers, pods, services, nodes, and a cluster itself. **Tools for Monitoring Resources** To scale an application and provide a reliable service, you need to understand how the application behaves when it is deployed. You can examine application performance in a Kubernetes cluster by examining the containers, pods, services, and the characteristics of the overall cluster. Kubernetes provides detailed information about an application's resource usage at each of these levels. This information allows you to evaluate your application's performance and where bottlenecks can be removed to improve overall performance. In Kubernetes, application monitoring does not depend on a single monitoring solution. On new clusters, you can use resource metrics or full metrics pipelines to collect monitoring statistics. Source: https://kubernetes.io/docs/tasks/debug/debug-cluster/resource-usage-monitoring/. A full metrics pipeline gives you access to richer metrics. Kubernetes can respond to these metrics by automatically scaling or adapting the cluster based on its current state, using mechanisms such as the Horizontal Pod Autoscaler. The monitoring pipeline fetches metrics from the kubelet and then exposes them to Kubernetes via an adapter by implementing either the `custom.metrics.k8s.io` or `external.metrics.k8s.io` API. Kubernetes is designed to work with OpenMetrics, which is one of the CNCF Observability and Analysis - Monitoring Projects, built upon and carefully extending Prometheus exposition format in almost 100% backwards-compatible ways. Source: https://kubernetes.io/docs/tasks/debug/debug-cluster/resource-usage-monitoring/.

U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	What is Kubernetes monitoring? Simply put, Kubernetes monitoring is the practice of tracking the status of all components of a Kubernetes environment. Because there are many pieces inside Kubernetes, Kubernetes monitoring actually entails monitoring many distinct things, such as: • The kube-system workloads. • Cluster information using the Kubernetes API. • Applications interactions with Kubernetes by monitoring apps bottom-up. Source: https://www.groundcover.com/kubernetes-monitoring. Further, Kubernetes monitoring includes providing alerts to problems occurring within a cluster that enables troubleshooting and addressing other potential issues. By collecting Kubernetes data, you'll get viable information regarding your Kubernetes cluster health, that can help you perform Kubernetes troubleshooting and manage issues like unexpected container termination. You can also leverage the data for proactive decisions such as adjusting rate limits. Source: https://www.groundcover.com/kubernetes-monitoring. As mentioned above, the Kubernetes API enables monitoring of the clusters. This can act as the main monitoring system and can monitor multiple Kubernetes clusters.

U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	Kubernetes alerting is the practice of generating notifications for events or trends in Kubernetes that require admins' attention. Examples of such events and trends include: • A node that has failed. • A Pod that is stuck in the pending state. • A container or Pod that is consuming a high level of resources relative to normal consumption trends. • High latency rates for communication between Kubernetes cluster components (such as between kubelet and control plane nodes). Source: https://www.groundcover.com/kubernetes-monitoring/kubernetes-alerting. Further, Kubernetes using API for cluster monitoring and kubelets for monitoring at node level, Kubernetes also supports additional third-party open source monitoring tools such as, Grafana, Prometheus, and/or FluentD.

U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	Setting up monitoring in Kubernetes Monitoring is a crucial aspect of effectively managing Kubernetes clusters. By setting up comprehensive monitoring, you can gain insights into your cluster's health, performance, and resource utilization, enabling you to address issues and optimize your infrastructure proactively. Several monitoring solutions are available for Kubernetes, including open-source tools like Prometheus, Grafana, and Alertmanager and commercial options like Datadog and New Relic. Choose a solution that aligns with your requirements and budget. Setting up Prometheus and Grafana for monitoring your Kubernetes cluster is a common choice and can provide a robust monitoring solution. The following are detailed steps for setting up Prometheus, Grafana, and FluentD, three common monitoring components, assuming that a Kubernetes cluster is already up and running: ▪ **Prometheus:** A monitoring tool built for aggregating metrics in containers that can be used for setting up alerts. ▪ **Grafana:** A monitoring tool that can be used to visualize Prometheus data. ▪ **FluentD:** A tool for aggregating a system's logs into one source. Source: https://www.apptio.com/topics/kubernetes/monitoring/.

U.S. Patent No. 7,822,841 (Claim 1)	
Patent Claim	**Example American Count II Systems and Services**
	 Kubernetes Cluster Monitoring Source: https://towardsdev.com/how-to-monitor-kubernetes-cluster-using-prometheus-grafana-f220b0284e65.